WEST Search History

Hide Items | Restore | Clear | Cancel

DATE: Thursday, July 15, 2004

Hide?	Set Name	Query	Hit Count
		DB=USPT,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=OR	
☐	L28	(server near5 (trigger$4 or caus$4) near5 client near5 boot$4)	3
☐	L27	(server near3 caus$4 near3 client near3 boot$4)	0
☐	L26	l20 and L25	0
☐	L25	L24 same (sav$4 near2 state)	15
☐	L24	(client same server same network$4)	26929
☐	L23	l10 and L20	9
☐	L22	l6 and L20	5
☐	L21	l1 and L20	16
☐	L20	l13 or l14 or l15 or l16 or l17 or l18 or L19	3296
☐	L19	713/340.ccls.	443
☐	L18	713/300.ccls.	856
☐	L17	714/15.ccls.	722
☐	L16	714/13.ccls.	263
☐	L15	714/7.ccls.	401
☐	L14	714/2.ccls.	667
☐	L13	714/1.ccls.	260
☐	L12	L6 and (sav$4 near3 state)	5
☐	L11	L5 and (sav$4 near3 state)	80
☐	L10	L4 same (sav$4 near3 state)	31
☐	L9	L5 same (sav$4 near3 state)	0
☐	L8	L6 same (sav$4 near3 state)	0
☐	L7	L6 same state	12
☐	L6	L5 same restor$8	28
☐	L5	L4 same configur$9	689
☐	L4	(reset$4 same boot$4)	2849
☐	L3	l1 same reset$4 same boot$4	0
☐	L2	(sav$4 near3 state) with ((prior or before) near3 fail$4)	12
☐	L1	(sav$4 near3 state) same ((prior or before) near3 fail$4)	48

END OF SEARCH HISTORY

L12: Entry 1 of 5 File: USPT Oct 2, 2001

DOCUMENT-IDENTIFIER: US 6298443 B1
TITLE: Method and system for supplying a custom software image to a computer system

Detailed Description Text (25):
The hard drive restoration process completes by clearing the information 226 on the hard drive 112 and downloading 228 the software image 102 from the custom-programmed CD ROM 106 to the hard disk drive 112. The restoration program clears the hard disk drive 112 by formatting the hard drive 112 to erase possibly corrupted data, insure proper operation of the drive, and eliminate any viruses that may have infected the drive 112. The restoration program downloads the software image 102 by copying all software originally ordered and configured from the custom-programmed CD ROM 106 onto the hard disk drive 112 in a correct order. As the filed are copied to the hard disk drive 112, file attributes are correctly assigned or reset for operation of the software image 102. The copy process is moderately time consuming, typically enduring for 10 to 25 minutes. Following copying of the files, a check software transport operation 230 execute; a routine that verifies that the software-hardware keying and software download were performed and executed correctly. The restoration program terminates 232 by displaying a message on the computer display requesting that the user remove the CD ROM from the reader, removing the bootable flexible diskette 108 from the drive, and rebooting the computer 104. Following the second reboot operation, the computer 104 is in the identical condition of the computer at the original delivery with the possible exception of differences resulting from any modifications made to the original software order by agreement between the user or customer and a factory representative.

Detailed Description Text (37):
In one mode of operation, the program code in the XBIOS 520 operates by transferring operation identifiers and parameters to the CMOS memory 460 and performing an input/output instruction that evokes a SMI# signal. The SMI# signal is a signal for activating a system management mode (SMM) of operating. When a processor 410 recognizes a SMI# signal on an instruction boundary, the processor 410 waits for all store operations to complete. The processor 410 then saves the processor register state to a region in memory called a system management RAM (SMRAM) space and begins to execute a SMM handler routine. The SMI# interrupt has a greater priority than debug exceptions and external interrupts so that SMM processing preempts debug and external interrupt conditions. Subsequent SMI# and nonmaskable interrupt (NM) requests are not acknowledged while the processor is operating in system management mode.

L12: Entry 2 of 5 File: USPT May 16, 2000

DOCUMENT-IDENTIFIER: US 6065123 A
TITLE: Computer system with unattended on-demand availability

Abstract Text (1):
A computer system with unattended on-demand availability includes power-saving features which place the system into a Standby mode whenever the system is idle or is not being used. Prior to entering Standby mode, the system sets a hardware timer which indicates when the next scheduled event in the system should be performed. When either the timer expires or another event occurs which requires system operation, the system resumes to the On power state without user intervention. In one embodiment, the system of the present invention allows applications to periodically save their operational states. By saving their operational states, applications are able to guard against power failures and crashes. If a power failure or crash occurs, the system consults restart policies and, if appropriate, automatically re-starts applications to their most recently saved operational states once power is re-stored.

Brief Summary Text (16):
In one embodiment, the system of the present invention allows applications to periodically save their operational states. By saving their operational states, applications are able to guard against power failures and crashes. If a power failure or crash occurs, the system consults restart policies and, if appropriate, automatically re-starts applications to their most recently saved operational states once power is re-stored.

Detailed Description Text (8):
In one embodiment, application processes are monitored by a server process that maintains information needed to restart them and restore their internal state, should it become necessary. By calling functions provided by the server process, applications can indicate the conditions under which they should be restarted, can save operational state information to be used upon restart, can schedule future execution of themselves or other applications, and can schedule messages to be delivered to themselves or other applications. By saving their operational states, applications are able to guard against power failures and software crashes. Furthermore, once the state information is saved, an application can schedule its future execution and then deliberately exit. When the scheduled time or other triggering event occurs, the application will be restarted and can recover its previous operational state and continue execution.

Detailed Description Text (45):
InstantON servicing agent 140 also checks whether an application or the system user has requested to enter Standby mode, step 325. This request may be direct from a system user or may be a procedure call issued from an application running on the system. In one embodiment of the present invention, the system user can request to enter Standby mode by switching the power system to the "off" position. In this embodiment, the traditional off/on power switch of a computer system is re-configured to cause the system to enter Standby mode rather than the Off power mode when the switch is placed in-the "off" position. Thus, in this embodiment the system does not turn "off"; therefore, if a power failure occurs, the system boots up as soon as power is restored. In one implementation, an additional power switch

is also included in the computer system which, when actuated, causes the system to enter the Off power mode. In an alternate embodiment, the system user can cause the system to enter the Standby power level by selecting a menu button provided by operating system 120 or InstantON manager 180, or by activating a predetermined key sequence, analogous to the ctrl-alt-del sequence used to reset some personal computers. The actuation of the predetermined key sequence by the system user is received by the operating system, which issues a signal to InstantON servicing agent 140, via VPOWERD 135, indicating the system is about to enter Standby mode.

Detailed Description Text (132):
In order to return an application to its operational state at the time of the system crash or power failure, a record is used to determine what that operational state is. In one embodiment of the present invention, this record is generated by the checkpoint services of the present invention. Applications which are connected to InstantON servicing agent 140 periodically make procedure calls to InstantON servicing agent 140 which save the necessary state information for the application in checkpoint records. Then, upon restart, InstantON servicing agent 140 provides restarted applications with the stored checkpoint information when the applications request it, thereby allowing them to return to their operational state preceding the system crash. In one implementation, this operational state is the state at the last time the checkpoint services were called prior to the system crashing.

Detailed Description Text (139):
The IonAddChkRecord procedure call allows the calling application process to generate a new checkpoint record. The calling process includes, as parameters, a pointer to a buffer and the size of that buffer. Prior to making the procedure call, the process generates the operational state information necessary for it to return to its current state. If this is the first time the process has called the checkpoint services, then all operational state information is saved. However, if this is a second or later call, then only incremental changes need to be included. The amount and nature of this checkpoint information being saved is dependent on the application process. Thus, the checkpoint records for each application process can be different. Upon receiving the IonAddChkRecord call, InstantON servicing agent 140 generates a checkpoint record for the process, storing the time the procedure was called, the size of the buffer and the data in the buffer. InstantON servicing agent 140 keeps track of all of the checkpoint identifiers associated with each application process based on the registration identifier for the process. Thus, multiple checkpoint records can be associated with the process regardless of whether the process was terminated and re-executed between the saving of the checkpoint records. In one implementation, InstantON servicing agent 140 returns a message indicating that the call was successful, that the calling process is not connected to InstantON servicing agent 140, that InstantON servicing agent 140 is unavailable, or that an invalid parameter was passed to InstantON servicing agent 140.

Previous Doc Next Doc Go to Doc#

First Hit Fwd Refs Previous Doc Next Doc Go to Doc#
Generate Collection Print

L28: Entry 2 of 3 File: USPT Mar 18, 2003

DOCUMENT-IDENTIFIER: US 6535976 B1
TITLE: Initial program load in data processing network

Detailed Description Text (29):
At step 500, the network administrator changes the setup of the server to specify a new hybrid RPL profile for a client. In other words, the `normal` mode of operation is disabled. At step 510, when the client system is next powered-on or rebooted (either using a remote control utility specified by the administrator or alternatively by the client user), the client issues one ore more NOS RPL requests via the network adapter. At step 520, the server recognizes the client unique network address in the request. However, in this mode, the server does not issue the hybrid bootstrap code to cause the client to execute a local boot. Instead, the server downloads selected software according to a software profile defined according to the required maintenance function. In the following description, this downloaded software is termed maintenance software though as will be described below it may in fact be software for upgrading the local operating system, software for upgrading system BIOS or other software. At step 530 therefore, the maintenance software is downloaded to the client system where it is executed and/or stored onto the local hardfile. At step 540, the client indicates to the server that the operation on the client is complete. In response, the server process changes the setup for the client back to hybrid RPL bootstrap at s step 550. At next reboot, indicated at step 560, which may be initiated either by the remote control utility or by the client user, the client issues the normal RPL request via the client network adapter card and local boot takes place as per normal hybrid RPL--step 570.

Previous Doc Next Doc Go to Doc#

L2: Entry 1 of 12 File: USPT Mar 16, 2004

DOCUMENT-IDENTIFIER: US 6708283 B1

**** See image for Certificate of Correction ****

TITLE: System and method for operating a system with redundant peripheral bus controllers

Detailed Description Text (21):

As discussed above, the I/O system manager 31 may periodically save the internal states of the selected peripheral bus controller 30 and the video controller 32. The system may also track the time between failures of the various controllers, or other statistics that allow the system essentially to predict controller failure. The system can then save the controller states, and change its selection of controllers at an appropriate time before a failure occurs.

L2: Entry 3 of 12 File: USPT May 31, 1994

DOCUMENT-IDENTIFIER: US 5317752 A
TITLE: Fault-tolerant computer system with auto-restart after power-fall

CLAIMS:

15. A method of operating a computer system having a central processing unit (CPU), memory including volatile memory and non-volatile memory, a main power supply, and a backup power supply, said method comprising the steps of:

(a) executing code by the CPU from the memory in normal operation while power for said computer system is supplied by the main power supply, said execution including controlling processes;

(b) detecting the occurrence of failure of said main power supply, and continuing execution of code by said CPU using the backup power supply;

(c) after detecting said failure, initiating execution of a shutdown procedure by said CPU, including issuing a sequence of signals from said CPU to said processes controlled by said CPU during normal operation immediately prior to said power failure, while continuing execution of said shutdown procedure by the CPU to save state of said processes being executed, the signals to said processes including:

(i) "signal power failure" (SIGPWR) with code "power failure quiesce" (PFQUIESCE) during shutdown followed by "signal power failure" (SIGPWR) with code "power failure restart" (PFRESTART), or

(ii) "signal terminated" (SIGTERM) with code "power failure quiesce" (PFQUIESCE) followed by "signal kill" (SIGKILL);

(d) storing on said non-volatile memory said state; and

(e) shutting down said backup power supply and ceasing execution of code by said CPU.

First Hit Fwd Refs Previous Doc Next Doc Go to Doc#

Generate Collection Print

L7: Entry 1 of 12 File: USPT Jun 8, 2004

DOCUMENT-IDENTIFIER: US 6748548 B2
TITLE: Computer peripheral device that remains operable when central processor operations are suspended

Brief Summary Paragraph Table (1):
Sleeping States Description S0 Normal operation, active state (no sleeping state). S1 The S1 sleeping state is a low wake-up latency sleeping state. In this state, no system context is lost (CPU or chip set) and hardware maintains all system context. S2 The S2 sleeping state is a low wake-up latency sleeping state. This state is similar to the S1 sleeping state except the CPU and system cache context is lost (the OS is responsible for maintaining the caches and CPU context). Control starts from the processor's reset vector after the wake-up event. S3 The S3 sleeping state is a low wake-up latency sleeping state where all system context is lost except system memory. CPU, cache, and chip set context are lost in this state. Hardware maintains memory context and restores some CPU and L2 configuration context. Control starts from the processor's reset vector after the wake-up event. S4 The S4 sleeping state is the lowest power, longest wake-up latency sleeping state supported by ACPI. In order to reduce power to a minimum, it is assumed that the hardware platform has powered off all devices. A copy of the platform context is written to the hard disk. S5 The S5 state is similar to the S4 state except the OS does not save any context nor enable any devices to wake the system. The system is in the "soft" off state and requires a complete boot when awakened.

Previous Doc Next Doc Go to Doc#

First Hit Fwd Refs Previous Doc Next Doc Go to Doc#

Generate Collection Print

L7: Entry 2 of 12 File: USPT Apr 13, 2004

DOCUMENT-IDENTIFIER: US 6721881 B1
TITLE: System and method for determining if a display device configuration has changed by comparing a current indicator with a previously saved indicator

Detailed Description Text (25):
FIG. 3 is a flow chart illustrating at least one embodiment of a method for detecting display configuration during an SBF fast boot. FIGS. 1 and 3 illustrate that, in operation 310, a system start is initiated when an initiating event is detected. An initiating event is any event that cycles the computer system 100 into a powered-on state. The reset event may be an initial supply of power to a computer system 100 that has been previously in a powered-off state, may be a user-initiated activation of a reset switch, may be a power cycle where power is removed and then restored to a computer system by a user or through an interruption of the power source, may be a software-initiated reset by the operating system, or any other event that causes, or emulates, the computer system's power transitioning from an "off" to "on" state. During the system start operation 310, the initiating event is detected, and execution of the system BIOS code 196 is begun, including execution of POST.

Previous Doc Next Doc Go to Doc#

L7: Entry 3 of 12 File: USPT Feb 10, 2004

DOCUMENT-IDENTIFIER: US 6691234 B1

**** See image for Certificate of Correction ****

TITLE: Method and apparatus for executing instructions loaded into a reserved portion of system memory for transitioning a computer system from a first power state to a second power state

Detailed Description Text (6):

FIG. 3 shows a state diagram illustrating the transitions of a computer system between various power states in accordance with ACPI specification. As mentioned above, the ACPI specification defines a number of global system states (Gx states) that apply to the entire system and are visible to the user. These various global system states include: (1) G0 global working state; (2) G1 global sleeping state; (2) G2 soft off state; and G3 mechanical off state. G0 working state is a computer state where the system dispatches user mode (application) threads and they execute. In this state, devices (peripherals) are dynamically having their power state changed. G1 sleeping state is a computer state where the computer consumes a small amount of power, user mode threads are not being executed, and the system "appears" to be off (from an end user's perspective, the display is off, etc.). Latency for returning to the working state varies upon the wakeup environment selected prior to entry of this state. Work can be resumed without rebooting the OS because large elements of system context are saved by the hardware and the rest by the system software. G2 soft off state is a computer state where the computer consumes a minimal amount of power. No user mode or system mode code is run. This state requires a large latency in order to return to the working state. The system's context will not be preserved by the hardware. The system needs to be restarted to return to the working state. G3 is a computer state that is entered and left by a mechanical means (e.g., turning off the system's power through the movement of a large switch, etc.) It is implied by the entry of this off state through a mechanical means that no electrical current is running through the circuitry. The OS must be restarted to return to the working state. There are various types of sleeping states within the global sleeping state. These various sleeping states include: (1) S1 sleeping state; (2) S2 sleeping state; (3) S3 sleeping state; and (4) S4 sleeping state. The S1 sleeping state is a low wake-up latency sleeping state. In this state, no system context is lost (CPU or chipset) and hardware maintains all system context. The S2 sleeping state is a low wake-up latency sleeping state except the CPU and system cache context is lost (the OS is responsible for maintaining the caches and CPU context). Control starts from the processor's reset vector after the wake-up event. The S3 sleeping state is a low wake-up latency sleeping state where all system context is lost except system memory. CPU, cache, and chipset context are lost in this state. Hardware maintains memory context and restores some CPU and L2 configuration context. Control starts from the processor's reset vector after the wake-up event. The S4 sleeping state is the lowest power, longest wake-up latency sleeping state supported by ACPI. In order to reduce power to a minimum, it is assumed that the hardware platform has powered off all devices. Platform context is maintained. From a user-visible level, the system can be thought of as being in one of the states shown in FIG. 3. In general use, the system alternates between the working states and the sleeping states. In the working state, the computer performs some work. User-mode application threads are dispatched and running. Individual devices can be in low-power states and the processor(s) can be in low power states if they are not being

used. Any device that is turned of by the system because it is not actively in use can be turned on with short latency. In one embodiment, when the computer system is idle or the user has pressed the power button, the OS will put the system into one of the various sleeping states shown in FIG. 3. No user-visible computation occurs in a sleeping state. The various sleeping states shown in FIG. 3 differ in what events can arouse the system to a working state and how long this takes. Computers that support legacy BIOS power management boot in the legacy state and transition to the working state when an ACPI OS loads. A system without legacy support transitions directly from the mechanical off state to the working state.

Previous Doc Next Doc Go to Doc#

First Hit Fwd Refs Previous Doc Next Doc Go to Doc#

Generate Collection Print

L7: Entry 5 of 12 File: USPT Sep 24, 2002

DOCUMENT-IDENTIFIER: US 6457137 B1
TITLE: Method for configuring clock ratios in a microprocessor

Detailed Description Text (15):
The term boot, as is well known in the art, refers to a process in a device designed to bring itself into a state where it can operate on its own. For example, a typical boot routine can consist of a small set of instructions that operate to start a computer by bringing the rest of a much larger process from a peripheral device into the memory for a processor from which the processor continues to execute. The small set of instructions typically resides in a read only memory (ROM) and the processor is configured to execute these instructions in response to a reset event. A reset event restores a computer or device to a known state. Typically, most devices will perform a power-on reset when power is introduced to the machine in order to initialize operation of the computer to a known state.

Previous Doc Next Doc Go to Doc#

L7: Entry 6 of 12 File: USPT Aug 6, 2002

DOCUMENT-IDENTIFIER: US 6430687 B1
TITLE: Boot sequence for a network computer including prioritized scheduling of boot code retrieval

Detailed Description Text (4):
In the preferred embodiment of computer network 100, it is typically unnecessary to fully power down network clients 104 during normal system operation.* During prolonged periods of inactivity, the preferred embodiments of network client 104 are configured to assume a low power mode to reduce the overall power consumed by computer system 100. In the preferred embodiment, network client can assume one of at least four power mode states. In an ON mode, network client 104 is fully functional and consumes the maximum power. If a client that is in ON mode remains idle (i.e., receives no input via a keyboard, mouse or other input device) from a user of network client 104 or from network server 102 via network 105 for specified duration, network client 104 assumes a SUSPEND state. In the SUSPEND state, various peripheral circuits and I/O facilities of network computer 104 such as the computer's video monitor are powered down until network client 104 detects user input or input from network server 102. For purposes of this disclosure, a defining characteristic of the SUSPEND state is the ability to "wake" network computer 104 from the SUSPEND state without executing a hardware or software boot. In other words, power is maintained to critical facilities of network client 104 to enable operation, in response to an input event such as a keyboard or mouse entry, after waking the I/O and peripheral circuits that were powered down. In the preferred embodiment, network computer 104 is capable of entering a SOFT OFF mode in which power is maintained to only those facilities of network computer 104 necessary to enable client 104 to detect a boot event and to initiate execution of a boot sequence. Whereas power and refresh activity is maintained to the network computer's system memory in SUSPEND sate, power to system memory is disabled in the SOFT OFF state. In embodiments of network client 104 lacking in a hard disk or other suitable permanent read/write storage facility, disabling power to the computer's system memory typically eliminates the network computer's operating system software from the system. Under these circumstances, execution of a boot code sequence is required after a subsequent boot event to restore network computer 104 to full functionality. In the SOFT OFF mode, a boot event that might suitably initiate the boot code sequence includes a LAN wake-up event in which the initiation of the boot code sequence and a user initiated boot event such as depressing a reset button on the chassis of network computer 104. In a FULL OFF mode, all power to network computer 104 is disabled and the computer is incapable of detecting any boot event other than the activation of a power switch located on network computer 104.

L7: Entry 7 of 12 File: USPT Jun 11, 2002

DOCUMENT-IDENTIFIER: US 6404741 B1
TITLE: Monitoring of a packet telephony device via a control device

Detailed Description Text (19):
One potential problem may arise if a full reset, or re-boot, of the packet telephony device is initiated. It is possible that the re-boot may retrigger the condition that had rendered the packet telephony device fully or partially inoperable. For example, the packet telephony device may be a PC-based packet phone which is also programmed to run other non telephony-related software, which software is the cause of the PC entering the hung state; re-booting the PC may cause the offending software to be executed again and re-create the hung scenario. As an alternative to attempting a full re-boot of the packet telephony device, a special command or series of commands can be issued, through device interface unit 240 or through power control handler 260, to initiate a limited restoration of operation of the packet telephony device so that the device manages only a limited set of functions. To. continue with the packet telephone example above, device interface unit 240 or power control handler 260 can issue a special command (or set of commands) which would initiate a limited re-boot of the PC, such that only telephony-related functions of the PC would be enabled. Alternatively, the packet telephony device could be configured such that any attempt to reset or re-boot that comes directly from power control handler 260 would result in the limited function reset described above.

L7: Entry 8 of 12 File: USPT Jun 4, 2002

DOCUMENT-IDENTIFIER: US 6401198 B1
TITLE: STORING SYSTEM-LEVEL MASS STORAGE CONFIGURATION DATA IN NON-VOLATILE MEMORY ON EACH MASS STORAGE DEVICE TO ALLOW FOR REBOOT/POWER-ON RECONFIGURATION OF ALL INSTALLED MASS STORAGE DEVICES TO THE SAME CONFIGURATION AS LAST USE

Detailed Description Text (20):
Typically, if ECP mode is selected, usually then the ECP channel would be selected, also enable and disable passwords such as the user password and the superior password, also set what the password should be, also determining if a password is required on boot, enable or disable the password on resume, store password protection for diskette of a floppy drive, fixed disk boot protection can be set to normal or write protected, enable the integrated hard drive interfaces, select primary integrated adapter, secondary integrated adaptor, both or disable, enable or disable the floppy disk controller, configure the serial port, disable, enable or auto, select disable, enable and auto for serial port configuration, select disable, enable and auto for infrared port configuration, select mode for infrared port or wireless port, IRDA or FIR, select the base I/O address for the infrared port, select the configuration of the parallel port to enable, disable or automatically configure the path by either the system BIOS or the operating system, select the mode of parallel port where the modes include normal, bidirectional ECP or EPP mode, select the configuration of the modem port to enable, disable or automatically configure the port by either the system BIOS or operating system, configure power management, configure the power management mode, always (power management for AC and battery power), battery only, disable (no power management), maximum performance to allow power conservation with optimal system performance, maximum power saving to allow most power saving at expense of system performance, custom, to allow custom setting for different power management features including smart CPU mode with off and on options, standby time out with disable and a predetermined period of time, suspend time out with disable and predetermined period of time, suspend with save to disk or suspend with save to RAM, resume; resume on modem, ring with enable or disable ring, resume on time of day, setting the time to set the resume time, battery low suspend with an enable or disable feature, inactivating timer, enable, disable, resume on alarm with enable or disable by setting the alarm time and alarm date, configure time-out function with disable with a fixed amount of time, stand-by time out, 5-Volt suspend time out, .0.-Volt suspend time out, hard disk time out, video time out, language, select primary IDE master, primary IDE slave, secondary IDE master, secondary IDE slave, all that is stored is what is found, select plug-in plug operating system including yes and no, reset configuration data including yes and no options, select system speed fast and compatible to set the speed of the memory cache, select error correction control (ECC) configuration, sets the memory ECC state including ECC or non-ECC, select resource configuration memory reservation to reserve specific memory blocks, IOQ to reserve specific IOQs, select keyboard configuration including NUM lock to set the power on state so that NUM lock is active or nonactive, select keyboard rate to select the keyboard repeat rate (in per sec), keyboard delay select delay before repeat of the keys, select video configuration palette snooping to enable or disable, DMI event logging including event log capacity, event log visibility, DMI event log data, clear the DMI event log, event logging disable or enable, mark the DMI events as read, select the setup password, restore on power loss to restore the last state before power loss occurred, stay

off to keep power off until power button is pressed, power on which restores power to the system, quick boot mode enable or disable to skip certain tests while booting.

Previous Doc Next Doc Go to Doc#

L7: Entry 11 of 12 File: TDBD Sep 1, 1993

DOCUMENT-IDENTIFIER: NB930925
TITLE: Method for Regeneration of Corrupted or Lost Binary Files Required for Booting

Disclosure Text (1):
Disclosed is a proposed method of recovering lost binary files that normally reside on the system disk by generating them from the system's boot image. Using a software tool, the boot image will be unpacked to extract copies of the binary files which originally created the boot image. These copies are then put back onto the system disk to replace lost or corrupted files. - The boot image is a conglomeration of several binary, executable files including the configuration files and the UNIX kernel. Essentially, the boot image is a micro-operating system containing enough information to get a system booted from the reset state. During boot up time, the resident Read Only Memory (ROM) will start reading a portion of the boot image into system memory. When enough of the binary has been read in, the boot image takes over and completes the system configuration and initialization. - A problem exists when an important file, such as the /UNIX* kernel file, gets corrupted or accidentally erased. The system can still boot up without the /UNIX file because a copy of that file has been copied into the boot image. Obviously, the user wants a copy of the /UNIX to reside on the system disks in case it is needed for construction of another, newer boot image. However, the recovery of that file is time consuming and troublesome with the current procedures. Currently, the user must go to either a backup medium or the original install media of the operating system. The user must use the media, most often tape, and scan the tape looking for the /UNIX file. Then, users must copy that file to the disk filesystem to recover the important file. A problem with this procedure, besides the time costs, is that there exists possibilities that the file recovered from the backup storage may not be the latest image that the system should be running. For example, if a new /UNIX were placed on a system and then the /UNIX file were accidentally erased, then restoring the /UNIX from the backup or install media would lead the user to restoring an obsolete version of the /UNIX binary. - This proposal calls for a software program that takes the boot image and unpacks the image, allowing the user to choose which binary files should be recovered. Since any update added to the system, that is essential to the booting of a machine, causes the boot image to be rebuilt, the boot image will have the latest binary copy of the essential files of an operating system. - The method of recovering critical binary files from the boot image improves many qualities of a system. First, the user has an easy way to recover lost or corrupted files. Second, the files that are recovered will be guaranteed to be the latest version of the file. Third, the regeneration recovery procedure will be less time consuming than the search and replace method. Finally, this feature reduces the dependency of having backup media present and available for recovery. * Trademark of UNIX System Laboratories, Inc.

First Hit Previous Doc Next Doc Go to Doc#

End of Result Set

Generate Collection Print

L7: Entry 12 of 12 File: DWPI Sep 25, 1993

DERWENT-ACC-NO: 1993-344325
DERWENT-WEEK: 199343

TITLE: Isolation barrier separating section contg. system configuration, power management and soft reset areas - is activated when VDP power supply level becomes invalid or drops below voltage level of back=up battery.

Basic Abstract Text (1):
The isolation barrier is incorporated in a CMOS integrated circuit chip to isolate a section, which is responsible for the system configuration, power management, soft reset and scheduling, from the rest of the chip. The isolation barrier is active when the VDD power supply becomes invalid (or when the VDD level drops below the voltage level of a back-up battery). The isolation barrier is powered by VDD and the isolated section is powered by either VDD or the back-up lithium battery VBAT, depending on the voltage level of VDD. When VDD is greater than VBAT, a signal called ISO ENA, which is an output of a VDD loss sensing circuit, is set to (logical) zero and the barrier is inactive, i.e. it lowers the barrier to establish a normal communication between the isolated section and the rest of the chip. When the voltage level of VDD drops below that of VBAT, the value ISOENA is set to (logical) one to enable the barrier. While the barrier is enabled, the system clock is shut down and the system management section is in idle mode but does not lose its contents. As ISOENA goes low and the barrier becomes disabled, the system clock restarts and the isolated section re-establishes communication with the rest of the chip. While the barrier is disabled, the memory and registers within the isolated section can be updated and read by the system. However when the barrier is enabled, the isolated section can only be accessed through a boot pin. Barrier enable mode: ISO ENA. The barrier control signal generated by a VDD loss sensing circuit, becomes active as VDD loss is detected. Active ISOENA signal stops the system clock as a pre-assigned phase and holds its outputs to the isolated section at an inactive (a logical zero) state, so that there will be no write/read access to the mere and registers in the section. As a result, the system information is maintained until the system is rebooted and power is restored again. Logically, the outputs from the barrier to the isolated section are generated from the inverse of the corresp. inputs, which are then NORed with the ISOENA signal. Thus, these output signals will be held at the logical zero state due to the active state of the ISOENA signal, whether the VDD power supply is present or not. If the VDD power supply is neither good or present the output signals from the barrier to the rest of the chip are useless and may be unknown. Barrier disable mode: ISOENA becomes inactive to lower the barrier to re-establish the normal communication between the isolated section with the rest of the chip.

Previous Doc Next Doc Go to Doc#

First Hit Fwd Refs Previous Doc Next Doc Go to Doc#

Generate Collection Print

L22: Entry 1 of 5 File: USPT Jun 8, 2004

DOCUMENT-IDENTIFIER: US 6748548 B2
TITLE: Computer peripheral device that remains operable when central processor operations are suspended

Brief Summary Paragraph Table (1):
Sleeping States Description S0 Normal operation, active state (no sleeping state). S1 The S1 sleeping state is a low wake-up latency sleeping state. In this state, no system context is lost (CPU or chip set) and hardware maintains all system context. S2 The S2 sleeping state is a low wake-up latency sleeping state. This state is similar to the S1 sleeping state except the CPU and system cache context is lost (the OS is responsible for maintaining the caches and CPU context). Control starts from the processor's reset vector after the wake-up event. S3 The S3 sleeping state is a low wake-up latency sleeping state where all system context is lost except system memory. CPU, cache, and chip set context are lost in this state. Hardware maintains memory context and restores some CPU and L2 configuration context. Control starts from the processor's reset vector after the wake-up event. S4 The S4 sleeping state is the lowest power, longest wake-up latency sleeping state supported by ACPI. In order to reduce power to a minimum, it is assumed that the hardware platform has powered off all devices. A copy of the platform context is written to the hard disk. S5 The S5 state is similar to the S4 state except the OS does not save any context nor enable any devices to wake the system. The system is in the "soft" off state and requires a complete boot when awakened.

Current US Cross Reference Classification (1):
713/300

Previous Doc Next Doc Go to Doc#

L22: Entry 2 of 5 File: USPT Feb 10, 2004

DOCUMENT-IDENTIFIER: US 6691234 B1

**** See image for Certificate of Correction ****

TITLE: Method and apparatus for executing instructions loaded into a reserved portion of system memory for transitioning a computer system from a first power state to a second power state

Detailed Description Text (6):

FIG. 3 shows a state diagram illustrating the transitions of a computer system between various power states in accordance with ACPI specification. As mentioned above, the ACPI specification defines a number of global system states (Gx states) that apply to the entire system and are visible to the user. These various global system states include: (1) G0 global working state; (2) G1 global sleeping state; (2) G2 soft off state; and G3 mechanical off state. G0 working state is a computer state where the system dispatches user mode (application) threads and they execute. In this state, devices (peripherals) are dynamically having their power state changed. G1 sleeping state is a computer state where the computer consumes a small amount of power, user mode threads are not being executed, and the system "appears" to be off (from an end user's perspective, the display is off, etc.). Latency for returning to the working state varies upon the wakeup environment selected prior to entry of this state. Work can be resumed without rebooting the OS because large elements of system context are saved by the hardware and the rest by the system software. G2 soft off state is a computer state where the computer consumes a minimal amount of power. No user mode or system mode code is run. This state requires a large latency in order to return to the working state. The system's context will not be preserved by the hardware. The system needs to be restarted to return to the working state. G3 is a computer state that is entered and left by a mechanical means (e.g., turning off the system's power through the movement of a large switch, etc.) It is implied by the entry of this off state through a mechanical means that no electrical current is running through the circuitry. The OS must be restarted to return to the working state. There are various types of sleeping states within the global sleeping state. These various sleeping states include: (1) S1 sleeping state; (2) S2 sleeping state; (3) S3 sleeping state; and (4) S4 sleeping state. The S1 sleeping state is a low wake-up latency sleeping state. In this state, no system context is lost (CPU or chipset) and hardware maintains all system context. The S2 sleeping state is a low wake-up latency sleeping state except the CPU and system cache context is lost (the OS is responsible for maintaining the caches and CPU context). Control starts from the processor's reset vector after the wake-up event. The S3 sleeping state is a low wake-up latency sleeping state where all system context is lost except system memory. CPU, cache, and chipset context are lost in this state. Hardware maintains memory context and restores some CPU and L2 configuration context. Control starts from the processor's reset vector after the wake-up event. The S4 sleeping state is the lowest power, longest wake-up latency sleeping state supported by ACPI. In order to reduce power to a minimum, it is assumed that the hardware platform has powered off all devices. Platform context is maintained. From a user-visible level, the system can be thought of as being in one of the states shown in FIG. 3. In general use, the system alternates between the working states and the sleeping states. In the working state, the computer performs some work. User-mode application threads are dispatched and running. Individual devices can be in low-power states and the processor(s) can be in low power states if they are not being

used. Any device that is turned of by the system because it is not actively in use can be turned on with short latency. In one embodiment, when the computer system is idle or the user has pressed the power button, the OS will put the system into one of the various sleeping states shown in FIG. 3. No user-visible computation occurs in a sleeping state. The various sleeping states shown in FIG. 3 differ in what events can arouse the system to a working state and how long this takes. Computers that support legacy BIOS power management boot in the legacy state and transition to the working state when an ACPI OS loads. A system without legacy support transitions directly from the mechanical off state to the working state.

Current US Original Classification (1):
713/300

Previous Doc Next Doc Go to Doc#